Implementation of Software Defined Networking for IoT using Virtualization

Anubhav Dinkar[1], Aniruddh M[2], Dr. Kiran V[3]

[1]Student, Dept of Electronics and Communication, R.V. College of Engineering, Bangalore, INDIA

[2]Student, Dept of Electronics and Communication, R.V. College of Engineering, Bangalore, INDIA

[3]Associate Professor, Dept of Electronics and Communication, R.V. College of Engineering, Bangalore, INDIA

[1]anubhavdinkar05@gmail.com, [2]animaruthesh@gmail.com, [3]kiranv@rvce.edu.in

Abstract: *Software Defined Networking (SDN) in an emerging networking architecture and IoT devices are ubiquitous in today's technologically developing world. In this work we use SDN to route the packets and simulate IoT devices using Virtual Machines. This work compares three encryption methods- AES-128, AES-256 and DES when they are employed for transmitting data between the IoT devices. A man in the middle attack is also simulated to validate the working of encryption methods. A feature is also provided to the user to use UDP sockets as opposed to the conventional TCP sockets. On comparing the algorithms, it can be determined that for large string lengths, AES-128 takes the shortest time to encrypt data.*

Keywords: **Virtual Machines, Amazon Lightsail, Linux, SSH, Networking, IoT, SDN, Encryption, Socket Connection**

1. INTRODUCTION

Software-Defined networking architecture that is dynamic, manageable, cost-effective, and adaptive, making it appropriate for today's high-bandwidth, dynamic applications. The network control and forwarding tasks are decoupled in this design, allowing network control to be directly programmable and the underlying infrastructure to be abstracted for applications and network services. The Internet of Things (IoT) is a term that refers to all devices that are connected to the internet. In this technologically growing world, the need to get real time information is also increasing at a rapid pace. That's where IoT devices come in- they are connected to the internet, be it a government owned module with a sensor array that continuously calculates the pollution level in a region or just a simple sensor that is used to get the temperature of the surroundings, if it is connected to the internet, it is an IoT device.

Since there are situations where IoT devices need to communicate with each other, it makes sense to have a secure connection between them. That is the premise of this work. If confidential information is being transmitted between IoT devices, it could be detrimental if it is intercepted. The only way to secure it is to encrypt the data before transmission, because even if it is intercepted, the attacker can't decipher the contents. In this work, we simulate 2 IoT devices that can communicate with or without encryption. We also simulate a man in the middle attack and compare the three protocols- AES-128, AES-256 and DES. All the networking is done using SDN.

2. LITERATURE REVIEW

In Al-Hayajneh et al [1], the authors have discussed the increase in usage of Internet of Things in various fields, such as wireless sensors, medical devices, home sensors, and so on. Due to this, security becomes a key aspect which is often overlooked due to the high number of possibilities and vulnerabilities in networking configurations. As IoT devices are based on the internet and can contain a lot of confidential information in its data transmission, it is important to focus on the security aspects of these devices. This paper implements a Software Defined Networking (SDN) based approach, due to its ease of configuration and also its efficiency in overcoming the shortcomings of traditional networking.
In Ramakrishnan et al [2], the authors have reiterated the growth of IoT in the network domain. Virtualization is a primary technology in the IoT domain and it is a major cause of the growth of IoT so rapidly. This paper visualizes the benefits and weaknesses related to the techniques employed in current IoT devices, while also discussing the ways in which virtualization can mitigate those issues in the future.

3. METHODOLOGY

In this work, with the help of the literature review as above, the IoT device construction was done with the help of Virtual Machines on Amazon Lightsail. These VMs act as a representation of the physical IoT device. A total of 3 VMs were created here, with one each being for the 2 IoT devices, and one being for a simulated Man in the middle, which we have used to simulate an actual case of a malicious attacker.

A. Changing the encryption algorithm in the socket

There are several encryption algorithms that exist in today's networking. Some of the more popular ones are RSA, AES and DES. In this work, the aim is to be able to change the encryption algorithm in the socket transmission so that the data, even if packet sniffing applications like Wireshark are used, is secure.

We simulated a communication between the IoT devices with the help of a socket connection, either using a simple unencrypted socket, and a socket encrypted with either of the above encryption algorithms. For this work, we have given the server side VM 4 options – AES-128, AES-256 and DES.

AES-128 is virtually unbreakable, and would take even the most powerful supercomputers several hundreds of years to brute force the key. While there are other methods like side channel attacks, this is only possible if the private data is somewhat related to the actual data that is to be transmitted. It is for this reason that in this work, the user has no control over choosing the key that is used for AES encryption; a random key is generated with the help of inbuilt OS libraries in Python. As a result of this, the RSA algorithm has to be used so as to provide a secure means to exchange the keys from server to client. It is to be noted that eventually, a JSON file is sent from the server to the client, mimicking the actual mode of transmission between any two IoT devices.
Further to AES-128, AES-256 uses a 32 bit key, which is also unbreakable, thereby providing that much more security to the data that is being transmitted. Finally, DES was one of the earliest encryption algorithms, which helped in advancements in the field, but it's 56 bit key is on the easier side to be broken. Hence, its use in modern cryptography has reduced.
In this paper, the Electronic Code Block cipher methodology is used in all 3 algorithms. It is easier because of direct encryption of each block of input plaintext and output is in the form of blocks of encrypted ciphertext.
A general schematic of the algorithm flow is shown in the following figures, which is applicable to all 3 algorithms.
The 3 VMs were installed with Amazon Linux operating system, while having defined IP addresses for all of them. These VMs were then accessed using the local Terminal (or command line interface) and the project was coded entirely in Python. All 3 VMs are kept in the same network for sake of ease and cleanliness of network management, but the idea of the project remains identical even if the devices were configured to be in different networks.

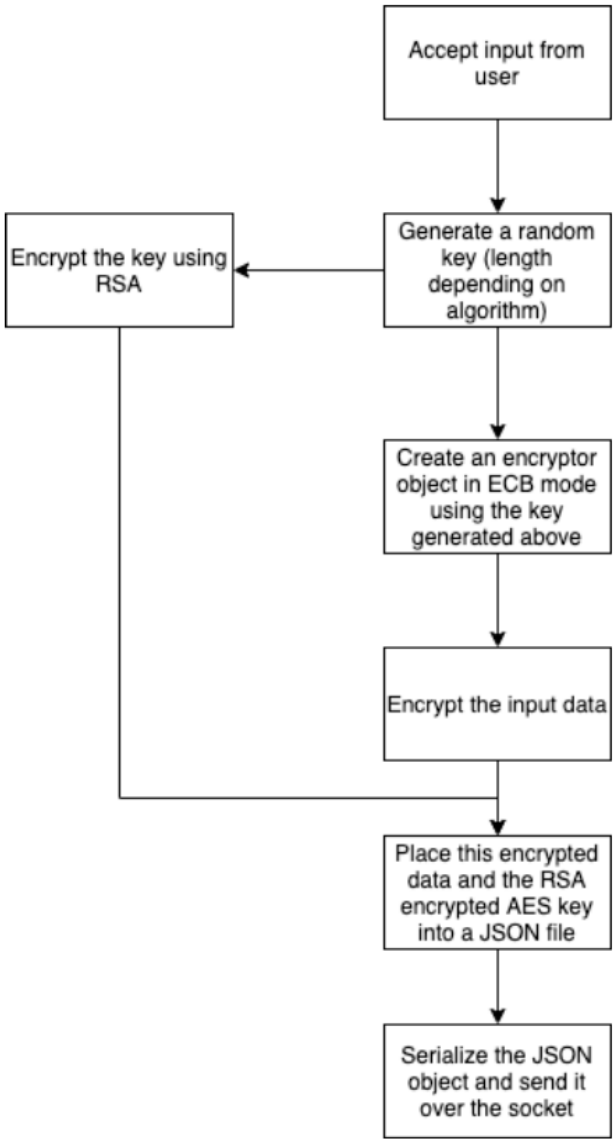

Fig. 1 The schematic of the encryption algorithm being used in this work

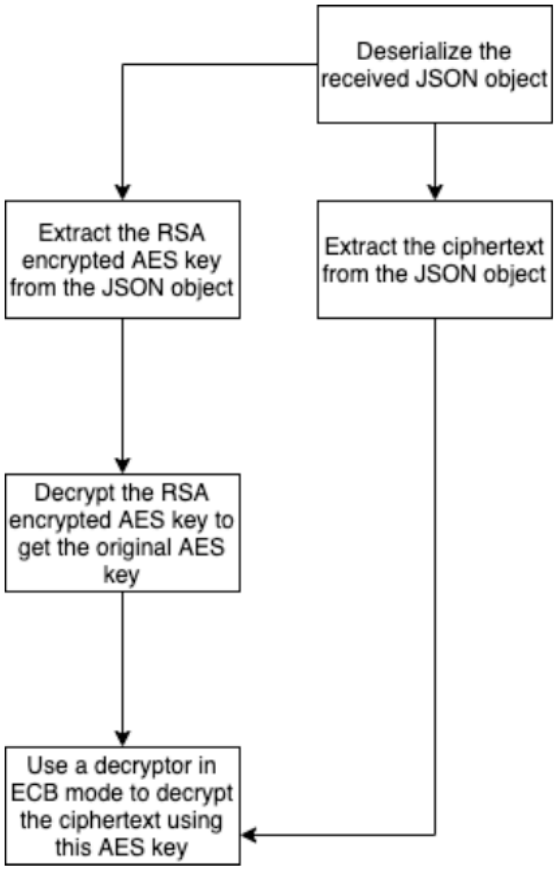

Fig. 2 The schematic of the decryption algorithm being used in this work

When an unencrypted mode of communication is chosen, the application windows in the Terminal look like this:

```
(.venv) [ec2-user@ip-172-26-12-82 server]$ python server.py
Enter Socket Type
1. TCP
2. UDP
1
waiting for connection
Welcome 3.7.49.16
Choose communication method:



1. Simple Text Exchange

2. Change Encryption Algorithm
1
hello
MITM Server Message Intercepted: hello
Client: Client: testing
MITM Client Message Intercepted: Client: testing
```

Fig. 3 Server Terminal Window

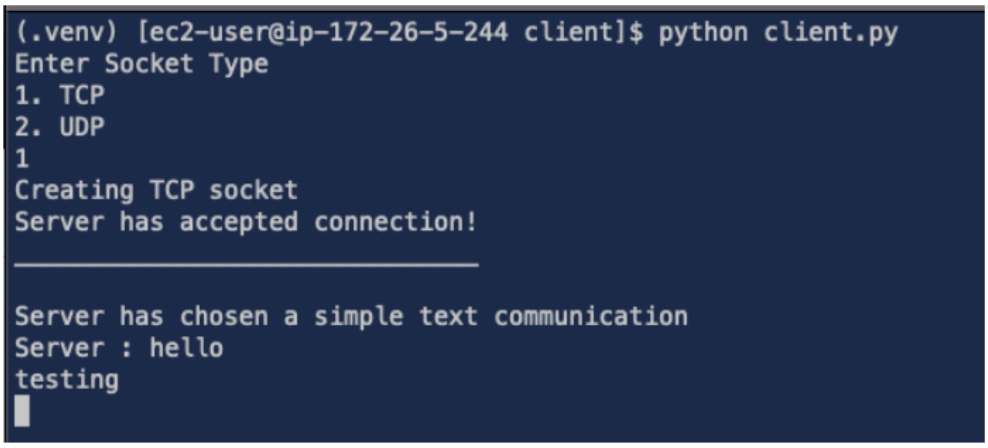

Fig. 4 Client Terminal Window

As we can see in the above screenshots, the communication is firstly established using a TCP protocol. This can be changed to UDP, which is explained in the further sections. Then, the server admin is responsible to choose whether encryption is to be done or not. In the above case, encryption is not chosen, thus we can see in Fig. 3 that the third man in the middle VM was easily able to sniff the packets being transmitted and obtain the transmitted data.

In case the server admin decides to encrypt the socket communication, the Man in the Middle attack will fail, as is seen in Fig. 5.

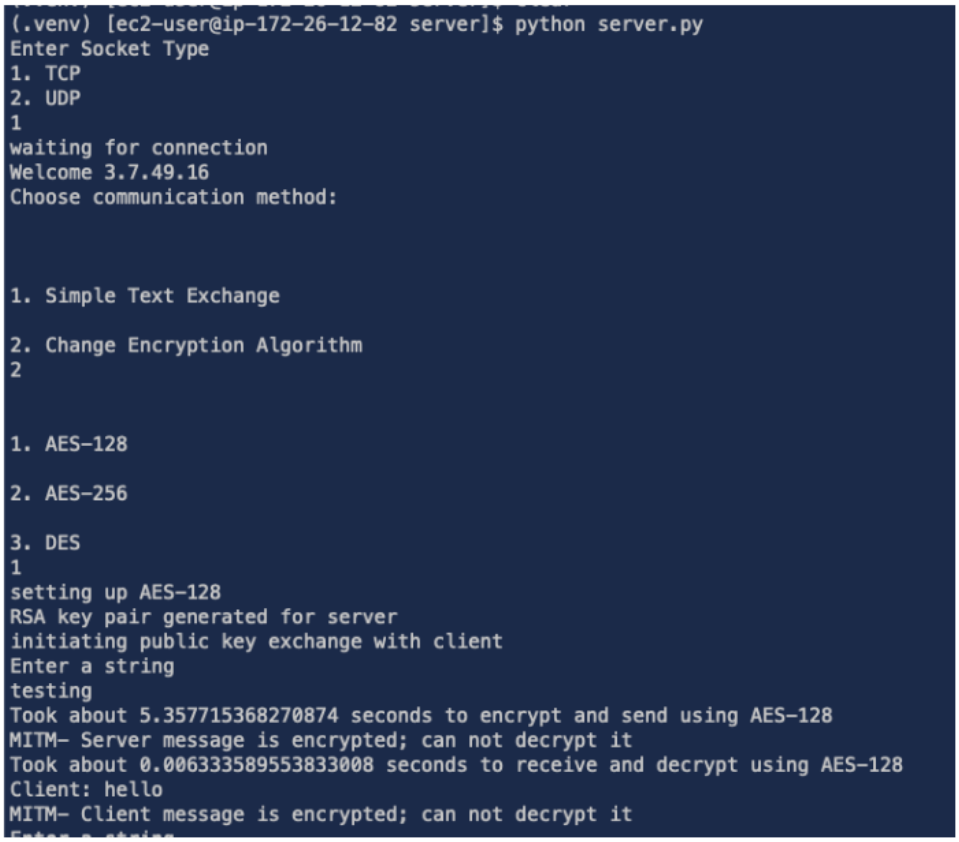

Fig. 5 AES-128 encryption algorithm being used by the server admin

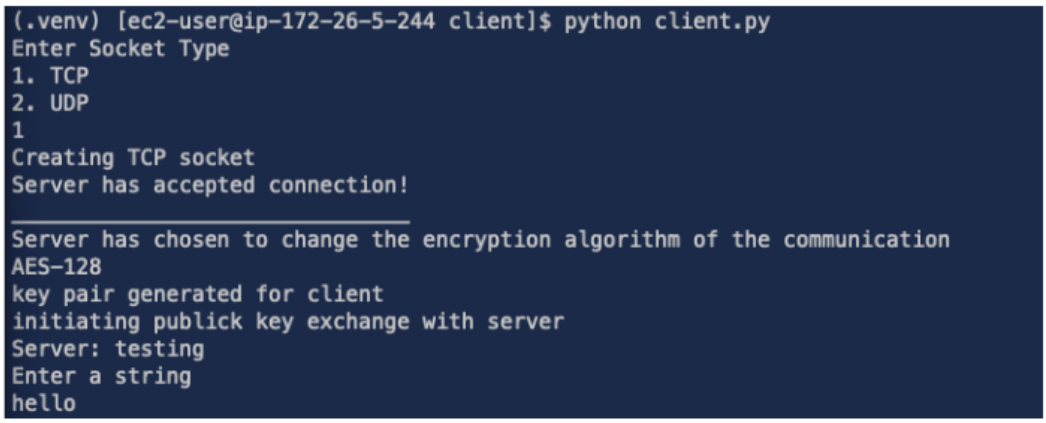

Fig. 6 Client side Terminal window with AES-128

This application also shows the time taken for each of the encryption and decryption operations being performed, which will be elaborated in the following sections.
As we can see above, and will be seen in the following figures, the first step before data transmission is done is to exchange the key between the client and server. It is once that key exchange is done successfully will the actual data transmission begin.

Similarly, the Terminal windows for AES-256 and DES are shown below:

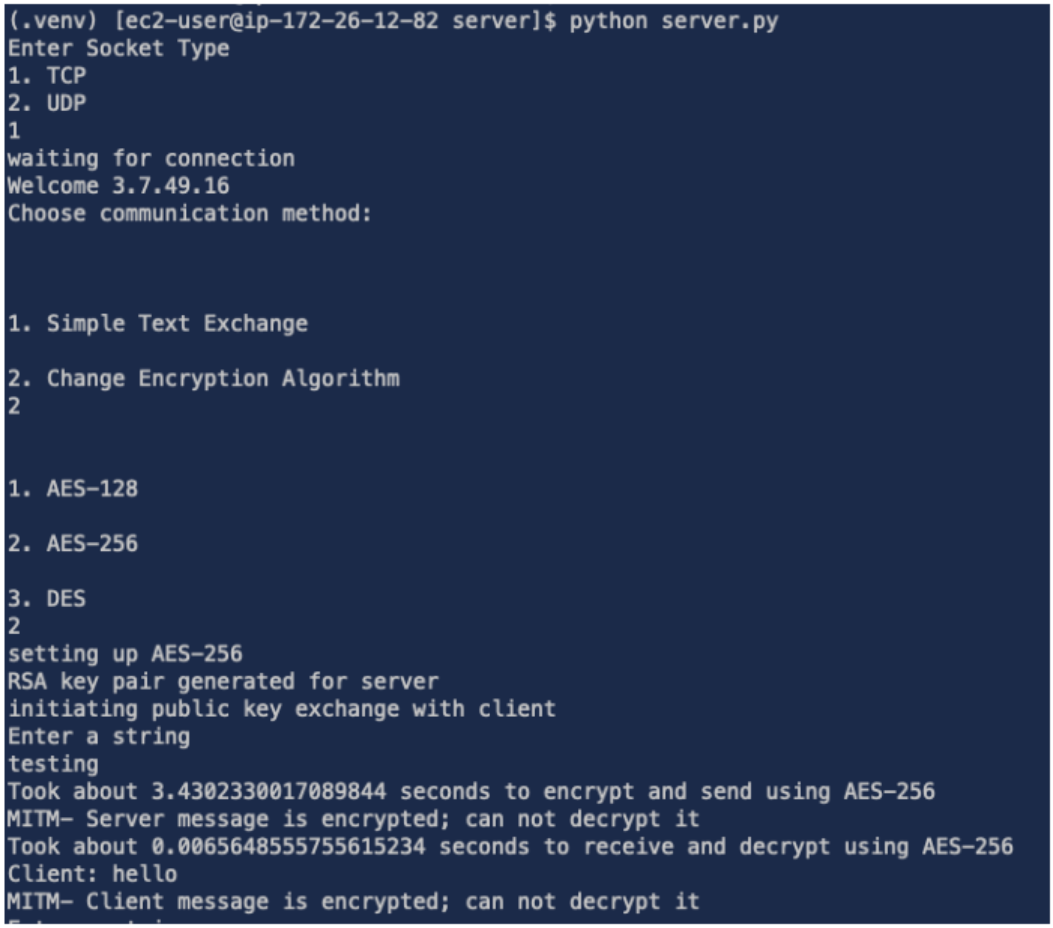

Fig. 7 AES-256 on the server side

```
(.venv) [ec2-user@ip-172-26-5-244 client]$ python client.py
Enter Socket Type
1. TCP
2. UDP
1
Creating TCP socket
Server has accepted connection!
_____________________________
Server has chosen to change the encryption algorithm of the communication
AES-256
key pair generated for client
initiating publick key exchange with server
Server: testing
Enter a string
hello
```

Fig. 8 AES-256 on the client side

```
(.venv) [ec2-user@ip-172-26-12-82 server]$ python server.py
Enter Socket Type
1. TCP
2. UDP
1
waiting for connection
Welcome 3.7.49.16
Choose communication method:



1. Simple Text Exchange

2. Change Encryption Algorithm
2


1. AES-128

2. AES-256

3. DES
3
Setting up DES
RSA keypair generated for server
Initiating public key exchange with client
Enter a string
testing
Took about 1.6754810810089111 seconds to encrypt and send using DES
MITM- Server message is encrypted; can not decrypt it
Took about 0.006455659866333008 seconds to receive and decrypt using DES
Client: hello
MITM- Client message is encrypted; can not decrypt it
```

Fig. 9 DES on the server side

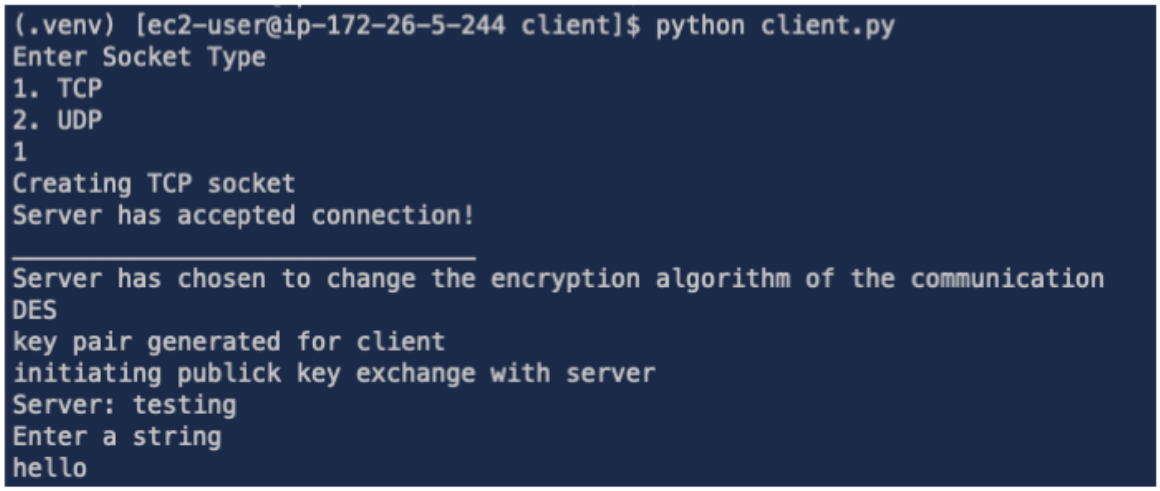

Fig. 10 DES on the client side

B. Changing the encryption algorithm in the socket

Currently, this work allows changing the protocol of the socket connection between TCP and UDP. This feature was also implemented as the size of the data packets as well as its kind may not be ideal for TCP, and UDP may achieve the transmission quicker. The implementations of the same are shown below.

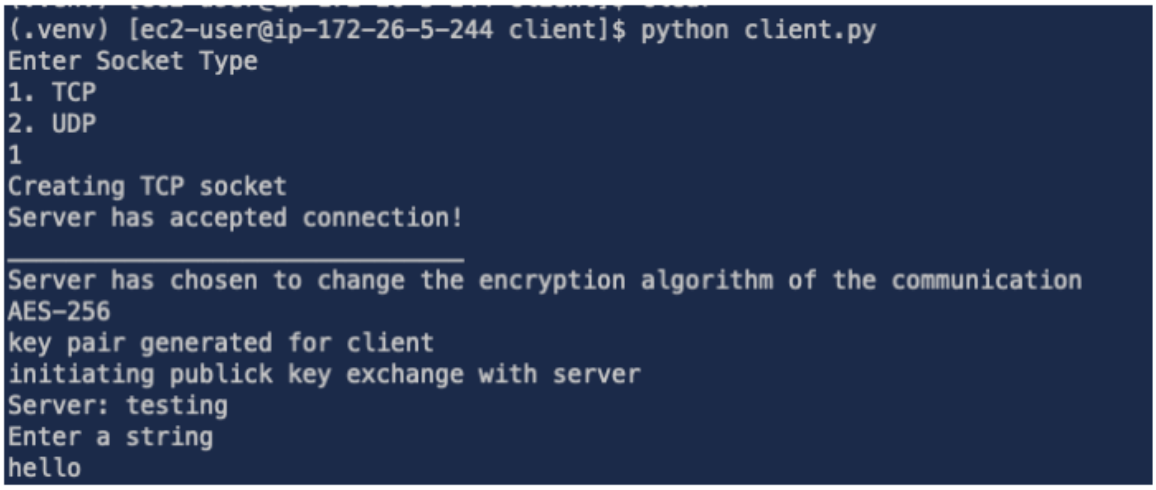

Fig 11. Creating a UDP Socket connection - Server side

Fig. 12 Creating a UDP Socket connection - client side

4. RESULTS

A. Changing the Encryption Algorithm

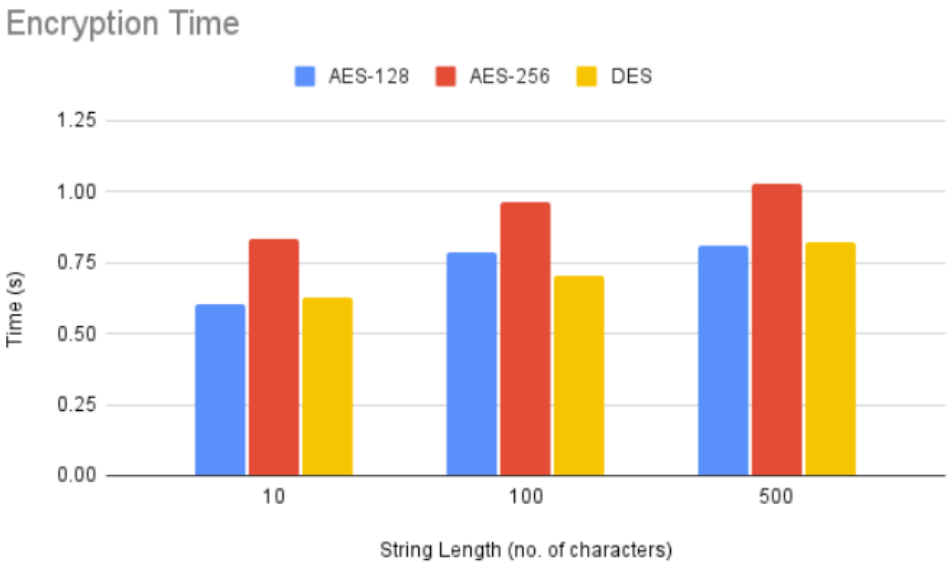

Fig. 12 Graph comparing the encryption times of the 3 algorithms

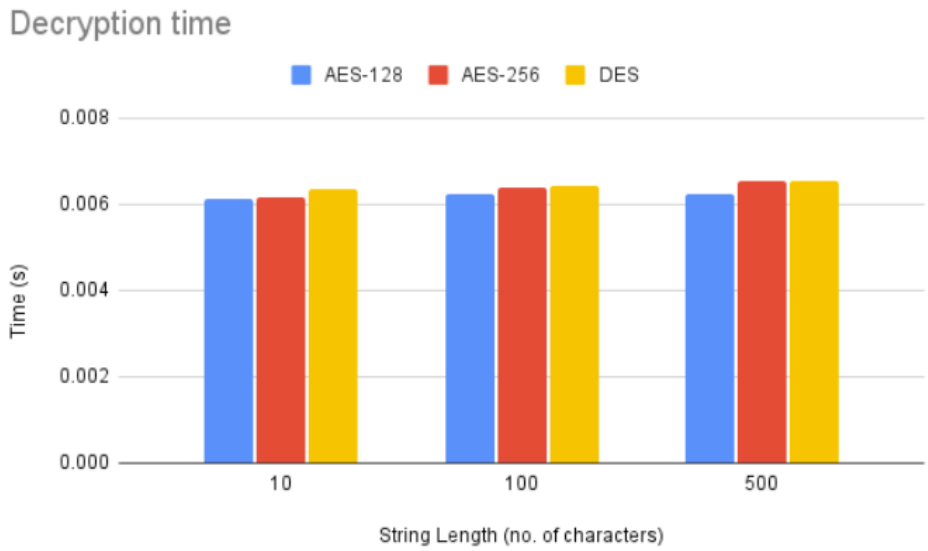

Fig. 13 The decryption times of the 3 encryption algorithms

It is evident from the encryption time that AES-256 takes the highest amount of time to encrypt data, while AES-128 and DES take about the same time, with the latter taking marginally more time. However, as the length of the text goes above 500 characters, DES starts to perform almost the same as AES-128, and in fact, the latter now starts to perform slightly better.

The decryption times are almost identical in all the 3 algorithms, but AES-128 here, too, seems to be the quickest mode of decryption, especially gaining some time over AES-256 as the length of the input data increases. DES performs worse than both the others in terms of decryption time, however the difference is negligible.

B. Changing the L4 protocol of the Socket

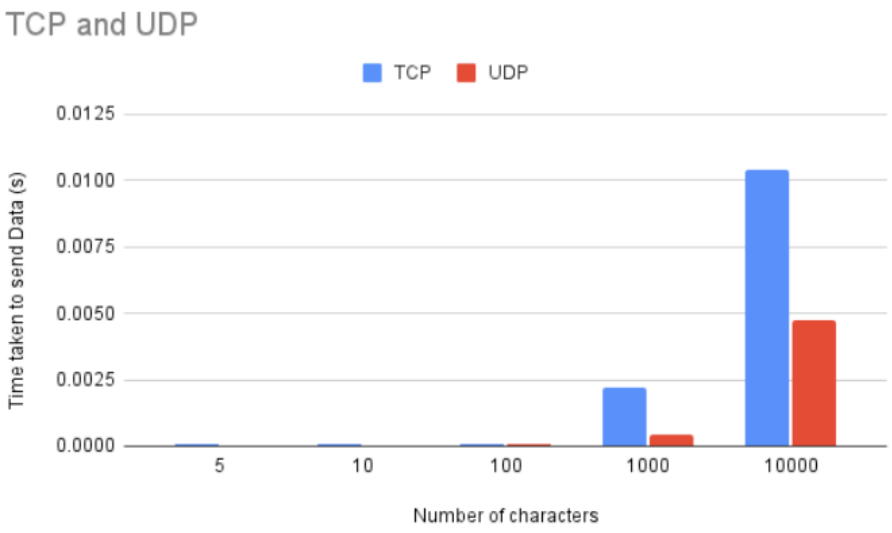

***Fig 14.* Comparing data transmission time between TCP and UDP Socket**

It is visible from the above graph that both TCP and UDP perform very similarly in the smaller text region, but when the number of characters increases to nearly 10000, TCP is much slower than UDP, almost by a factor of 2. It is to be noted that 10000 characters is only about 10 KB of data, which is still on the smaller side for real world data. Hence, UDP is shown to be better for this kind of data

5. CONCLUSIONS AND FUTURE SCOPE

In conclusion, we see that when any form of encryption is employed for a communication, it is almost impossible to decipher it. It is also observed that it takes relatively less time for AES-128 to encrypt large strings of data. It can also be noted that the man in the middle attack could not extract information when encryption was employed.

The future scope of this work is to migrate this to actual IoT devices and add more encryption choices. We could also add a feature to use SS. In addition to this, we can make the sender automatically change encryption algos based on the lengths of the strings chosen.

REFERENCES

[1] Al-Hayajneh, Abdullah, Zakirul Alam Bhuiyan, and Ian McAndrew. "Improving Internet of Things (IoT) security with software-defined networking (SDN)." Computers 9, no. 1 (2020): 8.

[2] Ramakrishnan, Jayabrabu, Muhammad Salman Shabbir, Normalini Md Kassim, Phong Thanh Nguyen, and Dinesh Mavaluru. "A comprehensive and systematic review of the network virtualization techniques in the IoT." International Journal of Communication Systems 33, no. 7 (2020): e4331.

[3] Daemen, Joan, and Vincent Rijmen. "AES proposal: Rijndael." (1999).

[4] Gan, G.; Lu, Z.; Jiang, J. Internet of Things Security Analysis. In Proceedings of the 2011 International Conference on Internet Technology and Applications, Wuhan, China, 16–18 August 2011; IEEE: 2011; pp. 1–4.

[5] Software-Defined Networking: The New Norm for Networks. Available online: https://www.opennetworking.org/images/stories/downloads/sdn-resou rces/white-papers/wp-sdnnewnorm.pdf (accessed on 5 January 2020)

[6] Tariq, J.; Riaz, T.; Rasheed, A. A Layer2 Firewall for Software Defined Network. In Proceedings of the 2014 Conference on Information Assurance and Cyber Security (CIACS), Rawalpindi, Pakistan, 12–13 June 2014; IEEE: 2014; pp. 39–42
[7] Abedinia O, Amjady N, Ghadimi N. Solar energy forecasting based on hybrid neural network and improved metaheuristic algorithm. Computational Intelligence. 2018;34(1):241-260.
[8] Verdouw CN, Wolfert J, Beulens A, Rialland A. Virtualization of food supply chains with the internet of things. J Food Eng. 2016;176: 128-136
[9] Sundmaeker H, Guillemin P, Friess P, Woelfflé S. Vision and challenges for realising the Internet of Things. Cluster of European Research Projects on the Internet of Things, European Commision. 2010;3(3):34-36
[10] Y. Zorian, "Ensuring Robustness in Today's IoT Era", Design & Test Symposium (IDT), 2015 10th International, 14-16 Dec. 2015, DOI: 10.1109/IDT.2015.7396723.
[11] F. Saffre "The Green Internet of Things", Innovations in Information Technology (IIT), 2015 11th International Conference on, 1-3 Nov. 2015, DOI: 10.1109/INNOVATIONS.2015.7381499.
[12] Q. Wang, W. Wang, K. Sohraby, “Energy Efficient Multimedia Sensing as a Service (MSaaS) at Cloud-Edge IoTs and Fogs”, IEEE Multimedia Communications Technical Committee (MMTC) Communications – Frontiers, Volume 11, Number 4, November 2016.
[13] A.H. Oti, E. Kurul, F. Cheung, J.H.M. Tah, "A Framework For The Utilization Of Building Management System Data In Building Information Models For Building Design And Operation", Automation in Construction, Elsevier B.V., Volume 72, Part 2, December 2016, Pages 195–210, DOI: 10.1016/j.autcon.2016.08.043.
[14] E. Z. Tragos, M. Foti, M. Surligas, G. Lambropoulos, et al, "An IoT Based Intelligent Building Management System For Ambient Assisted Living", Communication Workshop (ICCW), 2015 IEEE International Conference on, 8-12 June 2015, DOI: 10.1109/ICCW.2015.7247186.
[15] A. Corna, L. Fontana, A. A. Nacci, D. Sciuto, “Occupancy Detection via iBeacon on Android Devices For Smart Building Management”,